Operating System Support for Task-Aware Applications

João Garcia and Paulo Ferreira

Distributed Systems Group, INESC ID Lisboa, Portugal
www.gsd.inesc-id.pt

1 Introduction

All widely used operating systems, such as UNIX and Microsoft Windows, base their user interaction on the manipulation of applications and files. Users wanting to perform a particular task have to navigate the file system of one or more devices to find their data and may have to know which applications to use (and where to find it).

If users were freed from file and application management they would be allowed to interact with their computers simply by giving commands such as "I want to write report A", "I want to continue drawing component B" or even "I want to go back to what I was doing before this phone call". Better still, computers should know, based on the users previous activity which task, or at least which set of tasks, the user is interested in.

Providing a computer with task-awareness requires monitoring the computing and physical environment and trying to capturing user intent. Developments in ubiquitous computing and sensing technology have created an opportunity to move away from user interaction based on the file/application paradigm and introduce task-orientation as an alternative user interaction and data management paradigm.

The file/application paradigms hampers user activity because:

- Users generally view their work activity as a sequence of tasks
- Users should be provided with an execution context that is adapted to the task their are performing.
- Disk storage should be a transparent resource and not a binding form of data organization. For example, in contrast to file systems, computer users are constantly accessing primary memory but are however completely unaware of memory management.

To address these issues we have built an operating system component which lets applications interact with users using a task-centered model. We also show how this can be applied to a file explorer in order to make it truly task-aware.

2 Support for Task Awareness

We developed an operating system task-awareness module that monitors environment properties and associates the use people make of their computing devices (such as accessing files and applications) with their personal work tasks. It lets users and applications use tasks as first class concepts and aims to significantly reduce task management actions.

2.1 Relevant Environment Characteristics for Task Awareness

There are both computer and environment characteristics that are important to identify user task activity:

- Time is a main source of user activity patterns. People are known to do particular tasks on specific times of the day, week or month.
- Location: The same logic applies to location. There are certain jobs that people do at their offices while others are performed at home, etc...
- Communication: Knowing with whom a user is communicating and which device she is using may give away what the user is doing.
- GUI activity: Finally, detecting which application is being used or is accessing data also helps detect which task is being performed.

The task-awareness module monitors time, location (which device is being used) and the user's computing activity. Currently, communication has to be explicitly monitored at application level (see for example Fig. 2).

2.2 Architecture

The task-awareness module is composed of: i) an API; ii) a monitor of file system accesses, and process and GUI activity; iii) a log of file and task activity; iv) a registry of application callbacks; v) a repository of users' profiles and tasks.

Applications are provided with an API that allows them to manipulate (create, delete and rename) tasks and associate files, applications, processes or extensions with particular tasks. Applications register callbacks which include a set of notification criteria. Hence, they are notified when specific folders, processes, extensions or combinations thereof are accessed.

File system accesses monitoring is based on an Windows Installable File System (IFS) filter driver [1]. This IFS is inserted in the file system's stack (Fig. 1) and intercepts all operating system calls to the file system. It is the only component that is placed at kernel level. Naturally, only accesses that are relevant for existing callbacks are reported to the task-awareness module and then to applications.

File accesses are reported from the IFS to the task-awareness module. The module logs the access, checks if any application is interested in this particular access. If that is the case the relevant application is called back.

The task-awareness module maintains a time log of file accesses and task changes. It also manages the association of files to tasks. Task information is stored persistently in an XML file. If a user moves tasks and their files to another device, this file can be moved along with the tasks and updated at that location taking into account the place in that device's file system where the tasks where placed.

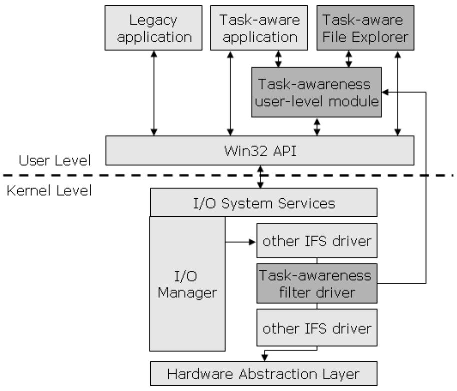
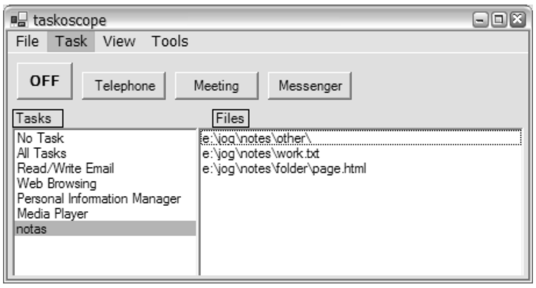


Fig. 1. Placement of the task-aware components within the Windows operating system

2.3 Task Explorer

The task explorer (Fig. 2) allows users to manage their work tasks and access their data accordingly. It uses the task-awareness module to manage tasks and get callbacks regarding file accesses. When it is unclear the user is asked which task a file belongs to. It is extremely difficult to eliminate user intervention in task management. The task explorer is a example of how to complement user activity monitoring with user guidance regarding the meaning of that activity. The aim of this explorer is that the user gradually does not have to say which task she is working on and consequently which files and applications she needs.



No Task
All Tasks
Read/Write Email
Web Browsing
Personal Information Manager
Media Player
notas

e:\jog\notes\other\
e:\jog\notes\work.txt
e:\jog\notes\folder\page.html

Fig. 2. Screenshot of the task explorer

The task explorer is configurable. It supports: the selection of which drives or folders to consider as storage location for task data; the association of particular tasks and applications (for example, whenever a user uses a RSS aggregator she is performing the task "Reading News"); and the selection of which extensions to be aware of (in order to exclude for example system and temporary files).

3 Related Work

The most relevant projects to have attempted to center user-computer interaction around tasks or activities are Activity-Based Computing (ABC, [2]) and the Aura project[3, 4]. Aura is geared towards moving a user's execution environment when she moves and choosing the best available applications for each file and/or task. ABC focuses on providing context aware information to doctors in a hospital setting. Another example is Cogenia [5], a system that provides users with location and device aware replication.

4 Conclusions & Future Work

Task-awareness simplifies users' interaction with computing devices. It is possible to use environment properties to detect which task a user is performing. Correctly and automatically identifying user tasks opens the door to simplifying many data management activities such as hoarding, staging and adaptation to devices.

In the future, we will develop applications that automate the kind of functionality proposed by Cogenia by using the task-awareness module's learning abilities to provide data replication in an automated and decentralized way.

Besides the detection of relevant environment characteristics, learning user activity patterns is important to develop task-awareness. We are developing a learning component within the task-awareness module that, based on a Bayesian network that is fed with users' task access patterns and context information, predicts which tasks a user is most likely to use in the future.

References

1. Nagar, R.: Windows NT File System Internals: a developer's guide. O'Reilly (1997)
2. Christensen, H.B., Bardram, J.: Supporting human activities – Exploring activity-centered computing. In: Fourth International Conference on Ubiquitous Computing UBICOMP 2002, Goteborg, Sweden, Lecture Notes in Computer Science 2401, Springer Verlag (2002)
3. Sousa, J.P., Garlan, D.: Aura: An architectural framework for user mobility in ubiquitous computing environments. In: 3rd IEEE/IFIP Conference on Software Architecture, Montreal (2002)
4. Garlan, D., Siewiorek, D., Smailagic, A., Steenkiste, P.: Project Aura: Towards distraction-free pervasive computing. IEEE Pervasive Computing, special issue on "Integrated Pervasive Computing Environments" **1**(2) (2002) 22–31
5. Rangan, M., Swierk, E., Terry, D.B.: Contextual replication for mobile users. In: International Conference on Mobile Business (ICMB'05). (2005) 457–463